https://mrcloudbook.com/open-source-project-implementing-devsecops-for-openai-chatbot-ui-deployment-devsecops/

https://github.com/mckaywrigley/chatbot-ui

https://www.youtube.com/watch?v=NgzETvHZUnU

Kubernetes-yaml file:

https://github.com/Aj7Ay/chatbot-ui/blob/legacy/k8s/chatbot-ui.yaml

**Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check**

**Install Plugin**

Goto Manage Jenkins →Plugins → Available Plugins →

Install below plugins

Blue ocean

1 → Eclipse Temurin Installer

2 → SonarQube Scanner

3 → NodeJs Plugin

4 → Docker

5 → Docker commons

6 → Docker pipeline

7 → Docker API

8 → Docker Build step

9 → Owasp Dependency Check

10 → Kubernetes

11 → Kubernetes CLI

12 → Kubernetes Client API

13 → Kubernetes Pipeline DevOps steps

sk-proj-t3osNZH\_scgIzpJuICbiRvPLNuiU-OqqhZk3dCnRm4xXVKbJJ8SrvqQuymGMXpRL0rOJJI6FsMT3BlbkFJbihghHCV63u3xr4B7ZzTPfn-i1b9sf5VpdgB\_CAV5apXauGlH0QMu3iAuWAehXy7dwc3T-vi0A

pipeline {

agent any

tools {

jdk 'jdk20'

nodejs 'nodeJS'

}

environment {

SCANNER\_HOME = tool 'sonar-scanner'

}

stages {

stage('Checkout from Git') {

steps {

git branch: 'legacy', url: 'https://github.com/Sravyatirumala/chatbot-ui.git'

}

}

stage('Install Dependencies') {

steps {

sh 'npm install'

}

}

stage('Sonarqube Analysis') {

steps {

withSonarQubeEnv('sonar-server') {

sh '''$SCANNER\_HOME/bin/sonar-scanner \

-Dsonar.projectName=Chatbot \

-Dsonar.projectKey=Chatbot'''

}

}

}

stage('Quality Gate') {

steps {

script {

waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'

}

}

}

stage('OWASP FS SCAN') {

steps {

dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'

dependencyCheckPublisher pattern: '\*\*/dependency-check-report.xml'

}

}

stage('TRIVY FS SCAN') {

steps {

sh 'trivy fs . > trivyfs.json'

}

}

stage('Docker Build & Push') {

steps {

script {

withDockerRegistry(credentialsId: 'Docker-creds') {

sh 'docker build -t chatbot .'

sh 'docker tag chatbot sravyatirumala/chatbot:latest'

sh 'docker push sravyatirumala/chatbot:latest'

}

}

}

}

stage('TRIVY') {

steps {

sh 'trivy image sravyatirumala/chatbot:latest > trivy.json'

}

}

stage('Remove Container') {

steps {

sh 'docker stop chatbot || true'

sh 'docker rm chatbot || true'

}

}

stage('Deploy to Container') {

steps {

sh 'docker run -d --name chatbot -p 3000:3000 sravyatirumala/chatbot:latest'

}

}

stage('Deploy to Kubernetes') {

steps {

script {

withKubeConfig(

credentialsId: 'k8s-cert',

clusterName: '',

contextName: '',

namespace: '',

serverUrl: '',

caCertificate: '',

restrictKubeConfigAccess: false

) {

sh 'kubectl apply -f k8s/chatbot-ui.yaml'

}

}

}

}

stage('k8s -trivy') {

steps {

script {

withKubeConfig(

credentialsId: 'k8s',

clusterName: '',

contextName: '',

namespace: '',

serverUrl: '',

caCertificate: '',

restrictKubeConfigAccess: false

) {

sh 'trivy k8s --report summary cluster > trivy-k8s.json'

}

}

}

}

}

}

 ……………………………………………………………………..

Click on openai.com to generate the API TOKEN

https://platform.openai.com/settings/organization/api-keys

Or we an generate from : https://rapidapi.com/

Create new secret key

Name: chatbot : sk-proj-psJ4lcBjd60qCsJY3WidyNUfa\_JAp29XyNdKwFZD16ufA7UmO15sfd1hhw8GCgj4RAzOg6g3yHT3BlbkFJM8SWybtkmZns9IlW2dRbrCCQR92kqUt3kSzF\_MqIaOLYcmrn-w-EPCQ-uMRqiDEHzWzqJzsfcA

Come back to chatbot UI that we deployed and bottom of the page you will see OpenAI API key and give the Generated key and click on save (RIGHT MARK)

……………………………………………………………………..  ……………………………………………………………………..

aws eks update-kubeconfig --name my-eks-cluster —region us-east-2

cd .kube

cat config

Save this in local and upload in below conf.

Go to manage Jenkins –> manage credentials –> Click on Jenkins global –> add credentials

Select Kind as Secret file and choose the file that you saved in your local for kubernetes configuration.

 ……………………………………………………………………..  ……………………………………………………………………..

https://github.com/Aj7Ay/chatbot-ui/blob/legacy/k8s/chatbot-ui.yaml

Give base code in this file and edit image name.

echo -n ' api’-key | base64

We will get base 64 code paste in this file.

 ……………………………………………………………………..  ……………………………………………………………………..

Create namespace:

kubectl create ns chatbot-ui

 kubectl get all

kubectl get svc